Published in proceedings of the International Conference on Advanced Computer Science and Information Systems (ICACSIS) 2017, Bali, Indonesia, 28-29 October 2017, which should be cited to refer to this work. DOI:10.1109/ICACSIS.2017.8355048

Assessing Data Veracity through Domain Specific Knowledge Base Inspection

Alex C. Olivieri *, Shaban Shabani † and Maria Sokhn ‡
Data Semantics Lab, Institute of Information Systems, HES-SO Valais-Wallis
Sierre, Switzerland
Email: *alex.olivieri@hevs.ch, †shaban.shabani@hevs.ch, ‡maria.sokhn@hevs.ch

Philippe Cudré-Mauroux
University of Fribourg
Fribourg, Switzerland
Email: pcm@unifr.ch

Abstract—**The Internet is nowadays a fantastic source of information thanks to the quantity of the information it provides and its dynamicity. However, these features also represent challenges when we want to consider trustworthy information only. On the Internet, the process of verifying information, known as *fact-checking*, cannot be performed by human experts given the scale of the information that should be manually checked, and the speed to which it changes. In this paper, we propose an approach to evaluate the trustworthiness of online information modeled as RDF Triples. Given a use case, we select a specific ontology (in the following we use movie reviews as a use case) and match its object properties with WordNet. This allows us to understand, for each input triple, which class the subject and the object belong to. We associate SPARQL queries to each class, which are then used by our approach to search for additional evidences in Wikidata. By doing so, our approach generates feature vectors that are used by machine learning classification models to predict the trustworthiness of new input triples. Experiments on real movie data show that our approach provides results that are on par or better than the state of the art in fact checking.**

I. Introduction

With the advent of semantic web technologies [1], new generations of Knowledge Management Systems (KMSs) have emerged [2]. These new KBMs are based on knowledge representation languages such as RDF [1], RDFS [2] and OWL [3] and divide the knowledge they express between a schema (Terminological Box - TBox) and instance data (Assertional Box - ABox) [12]. The Internet can be considered a gold mine for populating and evolving such knowledge bases (KB).

However, the information provided by the Internet is far from being trustworthy as its volume and speed are so high that it gets impossible for experts to verify manually all pertinent information. In order to exploit the knowledge provided by the Internet, new methodologies to automatically perform fact checking are needed. Verifying such facts involves two steps in the content of the new generation of semantic KBs: on one hand one must verify if the structure of the facts is coherent with the schema of the knowledge base while on the other hand one must verify the trustworthiness of the content of the information. Studies on the completeness of the TBox [3] and on the consistency between the ABox and TBox [4] are now mature and there exist tools such as reasoners that can detected logical inconsistencies easily. However, even though instance data represent the larger component of the information contained in semantics KBs, determining the veracity of instance data is still very challenging and requires more research.

The last few years have seen an increased focus on verifying the ABox, which is often considered as essential since it represents the vast part of the data that compose the Semantic Web. Previous approaches tend mainly to compare features of data expressing the same piece of information, such as their provenance, in order to understand which data are trustworthy [5], [7]. These approached are valid as long as there are multiple pieces of data about the same piece of information and it is possible to have some metadata which represent features attached to these data. But they have limitations when facts come alone and without additional metadata that can be used to evaluate their trustworthiness. Our hypothesis is that if we can embed the features needed to verify the information trustworthiness within the appropriate ontology on which the domain specific knowledge base is based, then we could also address the cases in which the previous approaches tend to fail.

In this paper, we propose an approach to assess the veracity of facts by enriching an ontology with some additional structures for supporting the creation of feature vectors of RDF triples. We do it by connecting Wordnet meanings, i.e. the specific senses for words, to the properties of the ontology and by providing to each class of the ontology meaningful sparql queries for Wikidata that will be used to query for additional evidence. The feature vectors are then used by a system leveraging machine learning classification algorithms to predict the trustworthiness of input facts.

Our research makes the following contributions: (1) it introduces a new data verification methodology tackling situations in which little information on the data is known; (2) our methodology allows to verify data originating from different domain just by changing the underlying ontology and defining queries on knowledge repositories that will be used to find additional evidence; (3) last but not least, the approach works when the knowledge base does not contain any instance data, thus it can be used to populate a knowledge base starting from zero.

The rest of this paper is organized as follows: in section II we present the use case that motivates our work and that

[1] https://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/
[2] http://www.w3.org/TR/rdf-schema/
[3] https://www.w3.org/TR/owl2-overview/

we use as a running example; in section III we describe the state of the art in assessing the trustworthiness of pieces of information; in section IV we describe the structure that we embed in the ontology for obtaining information from input RDF triples; in section V we describe the process to assess data veracity using machine learning algorithms; in section VI we explain the empirical tests that we performed, and discuss the results provided by our approach; in section VII we conclude the paper by summarizing our approach and listing some avenues for future work.

II. Motivating Scenario

Let us suppose that the Internet provides the following piece of information: "Will Smith acted in the movie Independence Day". If we ask a movie expert to provide us feedback about this information, he will probably say that it is trustworthy because this is a well known information. But what if the information is about a less known movie? Probably he would provide us with some feedback based on background knowledge and from asking himself questions such as "is the mentioned person an actor?" and "is there a movie with this title?". If both answers are yes, even if he is unsure about the trustworthiness of the information he can start to think about some positive answer.

This aforementioned information can be seen as a RDF triple having *Will Smith* as subject, *act* as predicate and *Independence Day* as object. Suppose that a knowledge base is based on an ontology which expresses the statement that the property *play* is a relationship between elements of the class *Actor* and elements of the class *Movie*. A system for verifying information has to perform four main tasks in order to predict if the triple expresses a trustworthy fact or not: a) understand that the predicate *act* has the same meaning of the property *play* for this domain; b) figure out if Will Smith is an actor; c) figure our if Independence Day is a Movie and d) understand if Will Smith actually played in Independence Day.

Thus, three main components are needed in the context to correctly answer the question:

1) An ontology that expresses the semantics of relationships because as one needs to know that *act* is a synonyms of *play* for the current domain.
2) Suitable repositories of knowledge which can be queried for evidences on doubtful pieces of information
3) A system which can make predictions, based on the evidence, about the veracity of input RDF triples about the movies domain.

III. Background

Information available on the Internet can be a big resource for knowledge bases. However, since anyone can publish information online, mechanisms for checking the veracity of the data are increasingly needed. There is a growing number of work that focus on verifying information in order to understand which pieces of data can be used to enrich knowledge bases.

Some approaches, regrouped under the term "data fusion", aim to identify the true values of data items provided by different data sources. [5] uses the generic knowledge base Freebase to obtain more information about data items and then applies machine learning models in order to predict which value is correct. [6] uses the same approach combined with indication about data sources reliability in order to create a large scale knowledge base. [9] uses Machine Learning algorithms to select the right value for a fact among many provided values. [7] provides a survey of techniques to fuse data. These approaches provide reasonable results, however they require redundant data for each piece of information, which is not always possible in practice.

[8] works on single piece of information by finding corroborating sources on the Internet in the form of textual occurrences. It then computes a confidence value from the various facts that are gathered. Like [6] and other approaches, it uses information coming from various sources, though this can lead to evaluation errors when wrong information is propagated as studied in [10].

[17] also works on a single piece of information at a time. It leverages a generic knowledge base, in this case DBPedia, represented as a knowledge graph and then studies how triples similar to the input triple are placed in the graph. It estimate a truth value depending on the distance in the graph between the classes of the subject and the object as input. One of its main limitation it that it performs well on well known categories of information but not in rare ones. This typically happens when leveraging generic knowledge base such as DBPedia or Freebase, which are not specific to the domain of interest.

IV. Defining the Structure for Creating the Features Vectors

Figure 1 illustrates the pipeline used to assess the veracity of the data in this paper. In phase 1 is performed the samples retrieval. In phase 2 the enriched ontology is used to created the feature vectors for every samples. In phase 3 machine learning classification models use the feature vectors to assess the data veracity. In this section, we first describe the structure needed to perform phase 2 of the pipeline, which is the module responsible for generating the features vector for a given input triple. After that, in the following section we describe in detail how the three phases are combined to perform as a single system.

A. OWL Domain Ontology

One of the foundations of our approach is to make use of additional knowledge a to verify the information. Section II introduced a scenario about movies. Following this, we selected the Movie Ontology - MO [4] as reference ontology, which provides a controlled vocabulary to semantically describe movie-related concepts and the corresponding individuals. We are interested only in its structural part (its TBox rather than its ABox) since we want to verify information by finding

[4] http://www.movieontology.org/

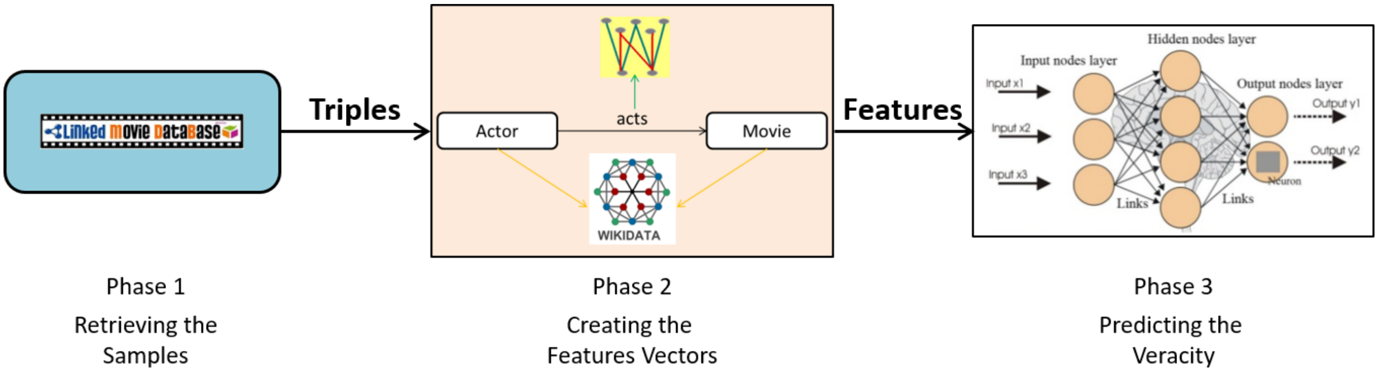


Fig. 1: The pipeline retrieves the samples for the experiments, transforms them in feature vectors of which one subset is used to train the predictive models, and the second one to evaluate the performance of assessing the trustworthiness of information

evidence in outer knowledge bases, without using the support that instance data could offer. Thus, we discard all RDF triples concerning the ABox from the MO. Moreover, we only use a subpart of the full ontology, the one describing the classes and properties we consider most important for our purpose.

B. Attaching Wordnet Synsets to Relationships

In OWL, a property is a relationship that connects a domain class with a range class. Between two classes there can be more properties and each of them expresses a different meaning. Figure 2 shows that the Actor and Movie classes are linked through two properties, one that goes from Movie to Actor and another one that goes through the opposite path. The property that goes from Actor to Movie expresses the following meaning: "an actor acts in a movie". It uses the active form of the verb *act*. The other one expresses the same meaning but using the passive form. Owl already manages such inverse relationships by association the *inverseOf* [5] property to a relationship, indicating that a property has the same meaning but with the domain and range switched. In our work, we want to make sure that we manage both the active and the passive form of a verb and that we can also manage the synonyms of the verb for the intended meaning.

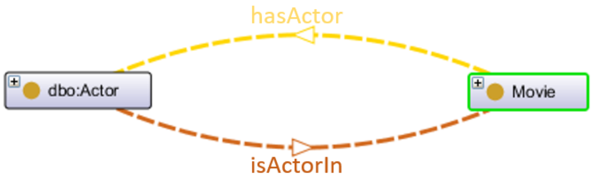


Fig. 2: Actor-Movie relationships

To provide meaning to the predicates of our ontology, we connect them to Wordnet [13]. "WordNet is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept." In Wordnet, "meanings" represent the sense of words and are expressed by *synsets*. By identifying for each predicate the correct Wordnet verb, its associated synset and its form we can provide it with the correct meaning for the intended domain. For our example we have the predicate *isActorIn* that corresponds to the Wordnet verb *act* with synset value *3* and form value *active*. Figure 3 shows how we extended to the basic ontology in order to connect verbs with Wordnet meanings.

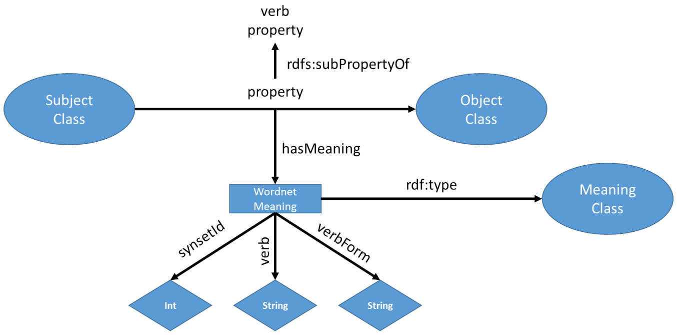


Fig. 3: Associating Wordnet Meaning to Relationships

To provide meanings to predicates we started by making the predicates sub-properies of the *verb* predicate. For the *verb* predicate we then provided an annotation property that links it to an instance of the class "Meaning". Each instance of the class meaning has three datatype properties, namely synsetId, verb and verbForm to express the Wordnet information. Table I shows the meaning provided to the property *isActorIn*. *px:* represents the prefix used in our ontology.

With this extension, it is now possible to understand for all new triples having a synonym of act as intended meaning that the subject is an instance of the class Actor and the object is an instance of the class Movie.

C. Connecting Classes to Wikidata

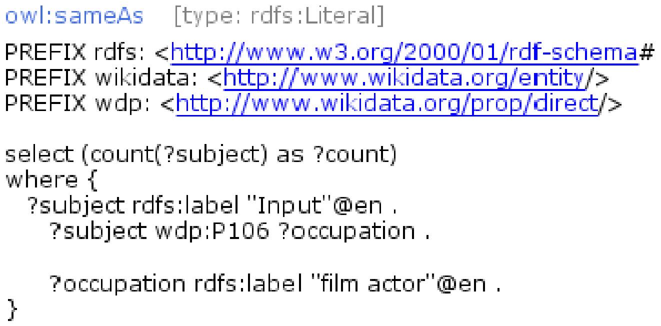
To evaluate the information conveyed by a triple, we have to find evidence that says that a subject and an object are effectively instances of the intended classes. Now that we can guess the belonging classes for the subject and the object of a triple, we have to find evidence that supports this guess.

[5] https://www.w3.org/TR/owl-ref/#inverseOf-def

TABLE I: Meaning on the predicate *isActorIn*

Subject	Predicate	Object
px:Actor	isActorIn	px:Movie
px:isActorIs	rdfs:subPropertyOf	px:verb
px:isActorIn	px:hasMeaning	px:act_1
px:act_1	rdf:type	px:Meaning
px:act_1	px:synsetId	3
px:act_1	px:verb	act
px:act_1	px:verbForm	active

We decided to use Wikidata [6] as knowledge base for finding additional evidence. For each class, we defined for it specific queries that can be run on the Wikidata Sparql end-point. For each query, Wikidata provides an answer that we transform in some feature for the feature vector of each input triple. Figure 4 shows the sparql query to find some evidence about if an *”Input”* is as instance of the Actor class in Wikidata. What the system does is retrieving the query from the ontology, substituting the field *”Input”* with the actor and check if the number of results of this query is zero or more than zero.



```
owl:sameAs   [type: rdfs:Literal]
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#
PREFIX wikidata: <http://www.wikidata.org/entity/>
PREFIX wdp: <http://www.wikidata.org/prop/direct/>

select (count(?subject) as ?count)
where {
  ?subject rdfs:label "Input"@en .
    ?subject wdp:P106 ?occupation .

    ?occupation rdfs:label "film actor"@en .
}
```

Fig. 4: Query for finding if an input is instance of the Wikidata Actor class

Table II shows the six queries for the triple *actor-act-movie*.

TABLE II: Queries for features

Query 1	Is the *Input (actor)* an instances of the Actor class?
Query 2	Is the *Input (movie)* an instances of the Movie class?
Query 3	Is the *Input (actor)* an instances of a superclass of the Actor class?
Query 4	How many connection has the *Input (actor)* with other instances of the Actor Class
Query 5	How many connection has the *Input (movie)* with other instances of the Movie class
Query 6	How many connection has the *Input (actor)* with other instances of the Movie class

V. Performing the Assessing of Data Veracity

In this section we describe the three phases that check the veracity of the data.

A. Retrieving Samples

To teach our model how to predict if a triple should be considered trustworthy or not, we must provide trustworthy (positive) and untrustworthy (negative) samples. In order to obtain a set of 10’000 positive samples, we queried the sparql endpoint of the semantic web database for movies LinkedMDB [7] by querying relationships of actors that played in movies.

Once we got the answers, we stored them in a file where each line represents a triple, e.g. *Will Smith/act/Independence Day* conveys the information of LinkedMDB *Will Smith as actor for the movie Independence Day*.

What we need now is to find meaningful negative samples that the model can use to figure out if an input triple is trustworthy or not. To do so, we queried again 10’000 relationships of actors that played in movies but excluding the previous relationships. Then we modified them in order to make them untrustworthy using the follow strategy: replacing subject, predicate, object, and combinations of them in a balanced way. In detail the following seven cases have been implemented.

1) Triples with correct actor and movie but wrong predicate
2) Triples with correct predicate and movie but replaced actor. We replaced actors 80% of the times with another instances of actors whom did not play the concerning movies and 20% of the time with instances of not actors.
3) Triples with correct actor and predicate but replaced movie. The same procedure (80% - 20%) is applied also here, as well as each time we replace actors and movies.
4) Triples with correct movie but wrong predicate and replaced actor.
5) Triples with correct actor but wrong predicate and replaced movie.
6) Triples with correct predicate but replaced actor and movie.
7) Triples with wrong predicate and replaced actor and movie.

By doing so, we created a set of 10’000 negative samples which balances the set of positive ones.

B. Features Creation

In order to train the machine learning system and to use it to make predictions, we first need to build features from the triples. For our use case (actors acting in movies) we decided to use 7 features that are describe hereafter. The first feature identifies if the predicate of a triple belongs to one of the object properties of our ontology which are sub property of the verb property, and the other six identify the answers of the queries defined in table II. For each sample, we now have a feature vector of 7 elements which contains the values of its features to which we added an 8th element - the label. Positive samples will be associated with a label = 1, while negative sample will

[6] www.wikidata.org

[7] http://www.linkedmdb.org/

be associate with a label = 0. Now we split both files in slices of 80% and 20% in order to create our *Training/CV Features Dataset* of 16’000 elements (8’000 positive + 8’000 negative) that will be used to train the system and *Test Features Dataset* of 4’000 (2’000 positive + 2’000 negative) elements that will be used to test how well our system makes predictions.

C. Machine Learning System Set-Up / Implementation?

To implement and evaluate our approach, we used the IPython framework [16], a tool for interactive scientific computing, and the Scikit-learn [8] toolkit which provides the necessary implementations of the machine learning algorithms. Moreover, we additionally made use of Pandas [9], a library for data processing, and Numpy [10] for numerical analysis.

First, we pre-processed the data. Since machine learning algorithms can speed up the training phase if all features have the same scale, we applied feature scaling and mean normalization to rescale the data into a range between 0 and 1. Subsequently, we chose appropriate machine learning algorithms for our problem. Because of the type of data and the number of samples, we choose the following machine learning classification algorithms: Logistic Regression, Random Forest, Support Vector Machine and Neural Networks.

Having the data preprocessed and algorithms in place, the next step was training the algorithms and running a 10-fold cross-validation, with data shuffling for each fold.

TABLE III: Accuracy of the models after CV

Algorithm	Mean	Std
Logistic Regression	78.29	1.17
Random Forest	77.11	1.06
Support Vector Machine	77.72	0.97
Neural Networks	78.29	1.17

Table III shows the mean accuracies provided by the cross-validation and the standard (Std) deviation between the 10 folds. The similarity in the performances indicates that we cannot choose one algorithm a priori, but we must proceed by using all of them for the testing phase.

VI. Tests, Results and Discussion

In this section, we describe the empirical tests we performed and the metrics we used, then discuss our results.

We run tests on the *Test Features Dataset*. The metrics for the results are precision, recall and their combination F1 score. Figure 5 shows the results of our tests for the four models. Neural network and Logistic Regression with their F1 score value of 0.86 perform better than the 0.843 of [8], which is the baseline for assessing data veracity on single piece of information. Furthermore, the F1 score value is not far from the 0.9 of [6] which exploits also metadata attached to input information.

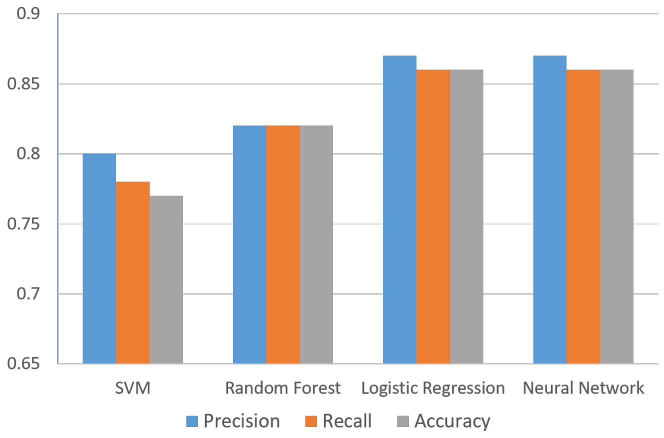


Fig. 5: Results on the Test Set

However, Neural Network and Logistic Regression provide identical values of precision, recall and F1 score, which led us to study the results further. We checked the precision, recall and F1 score for both class, positive and negative and we realized that this anomaly is only for the average values, since the singular class values are all different as showed in table IV.

TABLE IV: Results for Neural Network model

Class	Precision	Recall	F1 Score
0	0.93	0.77	0.84
1	0.80	0.95	0.87
Average	0.87	0.86	0.86

After studying and evaluating the results, we analyzed also the correlations between the various features to understand if we could eliminate some of them in order to speed up the models without losing in accuracy. Figure 6 shows that feature number 2 is highly correlated with feature number 4 and both of them do not seem to be really informative (see Table V). Therefore we run the models one time without using feature 4 and another time without using feature 2. The results obtained after these new configurations showed a decrease in precision and recall, which was especially pronounced for neural networks. Thus we decided to keep using all features.

TABLE V: Features Importance

f1	f2	f3	f4	f5	f6	f7
0.541	0.009	0.039	0.008	0.036	0.234	0.133

For our problem, recall and precision do not have the same significance for both classes. In fact, for the positive class (trustworthy triples) recall is more important than precision, since it shows that the model can figure out a high percentage of trustworthy triples. Contrarily, for the negative class precision is more important since it shows that the model confuses

[8] http://scikit-learn.org/

[9] http://pandas.pydata.org/

[10] http://www.numpy.org/

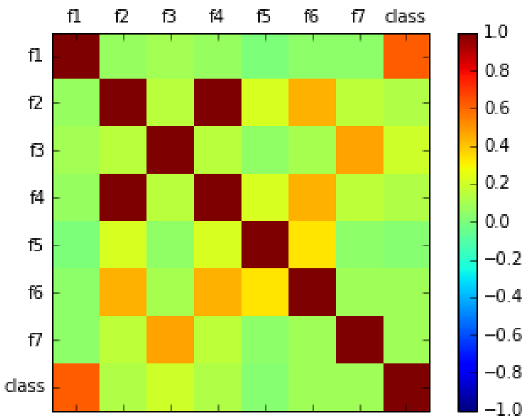


Fig. 6: Features Correlation

untrustworthy triple for trustworthy ones with a low percentage. This means that it would be possible to automatically enrich a knowledge base with a lot of trustworthy triples and with few untrustworthy ones. Figure 7 shows the results of each model with respect to this consideration.

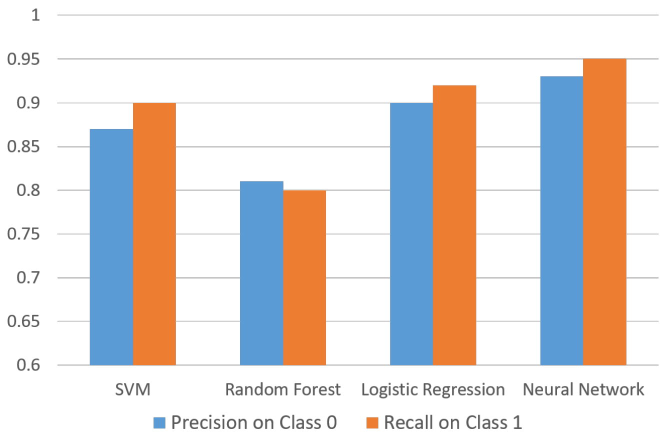


Fig. 7: Precision for class 0 and Recall 1 for class

It can be noticed that neural network performs better because the values of its precision on class 0 and recall on class 1 are really good and higher than the values of the other models. As a consequence, the neural network model allows to insert into a knowledge base a higher magnitude of new trustworthy information and fewer untrustworthy ones, which corresponds with the purpose of our approach.

VII. Conclusions and Future Work

We can conclude the paper by noting that our hypothesis of enriching a knowledge base with additional information about the domain of interest can improve the data veracity assessment task. We claimed in section I that our work provides three main contributions: a) tackling situations where little information about the data is known, b) capability of verifying data in different domains and c) populating knowledge bases from zero. The work proved that these claims are fulfilled. In fact, we can observe that our approach requires only to know the domain of the triples to be verified in order to select a meaningful ontology to support the data verification. No metadata are then required for the triples. Moreover, such an approach can be used in any scenario by using the appropriate ontology, adding the right Wordnet meanings and defining domain specific queries for the suited external repositories of knowledge. Furthermore, our approach does not require any prior ABox instances, thus it can be used for populating a knowledge base from scratch.

As future work, we plan to proceed in two directions. The first one is focusing on fine tuning the models we used in order to see if we can improve our results further and then to do a comparative evaluation with state-of-the-art approaches. The second one is to combine our approach with some related work [9], [17] and [6] in order to obtain a new, hybrid solution that provides the best of both worlds.

References

[1] Berners-Lee, Tim, James Hendler, and Ora Lassila. ”The semantic web.” Scientific american 284.5 (2001): 28-37.

[2] Davies, John, Miltiadis Lytras, and Amit P. Sheth. ”Guest Editors' Introduction: Semantic-Web-Based Knowledge Management.” IEEE Internet Computing 11.5 (2007): 14-16.

[3] Cord, Valentina, and Viviana Mascardi. ”Checking the completeness of ontologies: a case study from the semantic web.” Proc. of the CILC04 Workshop. 2004.

[4] Mendel-Gleason, Gavin E., Rob Brennan, and Kevin Feeney. ”Ontology Consistency and Instance Checking.”

[5] Dong, Xin Luna, et al. ”From data fusion to knowledge fusion.” Proceedings of the VLDB Endowment 7.10 (2014): 881-892.

[6] Xin Don, et al. Knowledge vault: a web-scale approach to probabilistic knowledge fusion. KDD 2014: 601-610

[7] Li, Xian, et al. ”Truth finding on the deep web: Is the problem solved?.” Proceedings of the VLDB Endowment. Vol. 6. No. 2. VLDB Endowment, 2012.

[8] Jens Lehmann, et al. DeFacto - Deep Fact Validation. International Semantic Web Conference (1) 2012: 312-327

[9] Yin, Xiaoxin, Jiawei Han, and S. Yu Philip. ”Truth discovery with multiple conflicting information providers on the web.” IEEE Transactions on Knowledge and Data Engineering 20.6 (2008): 796-808.

[10] Dong, Xin Luna, Laure Berti-Equille, and Divesh Srivastava. ”Integrating conflicting data: the role of source dependence.” Proceedings of the VLDB Endowment 2.1 (2009): 550-561.

[11] Olivieri, Alex Carmine. Improving Automated Fact-Checking Through the Semantic Web. ICWE (2016).

[12] Franz Baader, et al. The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press 2003, ISBN 0-521-78176-0

[13] Christiane Fellbaum (1998, ed.) WordNet: An Electronic Lexical Database. Cambridge, MA: MIT Press.

[14] Provost, Foster, and Ron Kohavi. ”Guest editors' introduction: On applied research in machine learning.” Machine learning 30.2 (1998): 127-132.

[15] Kluyver, Thomas et all, Jupyter development team, (2016) Jupyter Notebooks a publishing format for reproducible computational workflows Loizides, Fernando and Scmidt, Birgit (eds.) In Positioning and Power in Academic Publishing: Players, Agents and Agendas. IOS Press., pp. 87-90. (doi:10.3233/978-1-61499-649-1-87).

[16] Fernando Prez, Brian E. Granger, IPython: A System for Interactive Scientific Computing, Computing in Science and Engineering, vol. 9, no. 3, pp. 21-29, May/June 2007, doi:10.1109/MCSE.2007.53. URL: http://ipython.org

[17] Ciampaglia, Giovanni Luca, et al. ”Computational fact checking from knowledge networks.” PloS one 10.6 (2015): e0128193.